DOCUMENT-IDENTIFIER: US 20020091636 A1

TITLE: CAPTURING QUALITY OF SERVICE

---------- KWIC ---------

Application Filing Date - APD (1):
19990325

Brief Description of Drawings Paragraph - DRTX (2):
[0004] FIG. 1 is a block diagram of a server running an accounting application **monitoring** a network.

Brief Description of Drawings Paragraph - DRTX (20):
[0022] FIG. 26 is an illustration of the format of an ICMP error **reporting** message header and datagram prefix.

Brief Description of Drawings Paragraph - DRTX (23):
[0025] FIGS. 29A-29B are diagrams depicting a protocol independent, packet loss detection **monitor**.

Detail Description Paragraph - DETX (7):
[0034] The accounting process 14, as will be described in FIG. 2, has a core process that can handle data records from each of the equipment types above, as well as other equipment types, and can provide data to each of the plurality of user-defined data consuming applications. This accounting process 14 provides flexibility in choosing data consuming applications that use accounting data. Such applications can include billing, enterprise charge-back or cost allocations, capacity planning, trending, application **monitoring**, user profiling and so forth.

Detail Description Paragraph - DETX (12):
[0039] The data collectors 52a-52g are distributed throughout the network. The data collectors 52a-52g are placed close to or within the network device that the collector is assigned to. That is, the data collector can be in-line or out-of-line relative to the device **monitored**. The data collectors 52a-52g can be anywhere. The data collectors 52a-52g can be completely uncoupled from the devices except for communication paths. As new network devices 12 are added to the accounting support arrangement 10, new data collectors are also deployed.

Detail Description Paragraph - DETX (20):
[0047] The data can also be used by other applications such as network

planning, security, auditing, simulation, flow profiling capacity planning and network design and so forth. Thus, the Internet service provider can independently **monitor** and evaluate network traffic caused by remote employees and mobile users, for example.

Detail Description Paragraph - DETX (23):

[0050] Referring now to FIG. 4, a similar access configuration 100', as the configuration 100 (FIG. 3) can be used with an Extranet switch 122. Extranet access allows remote users to dial into an Internet service provider (ISP) and reach a corporate or branch office via an ISP. The Extranet switch allows Internet users access to corporate **databases,** mail servers and file servers, for example. It is an extension of the Internet in combination with a corporate Intranet. In this configuration, the Extranet switch 122 can be owned and operated by an Internet service provider as shown with enterprise A, or it could, alternatively, be owned and operated by an enterprise, as shown with enterprise B. Users would access the corporate network of either enterprise A or enterprise B, via the Internet service provider with various types of tunneling protocols such as L2TP, L2F, PPTP or IPSec, and so forth. The accounting server 13 located at the service provider and also accounting servers 13', 13" within enterprise A and enterprise B allow each the Internet service provider and each of enterprises A and B to run accounting process 14', 14" on the servers 13', 13" to **monitor** and collect network data.

Detail Description Paragraph - DETX (26):

[0053] In this case of the host connected to the port, or a router or some other device being connected to the port, there is no other connection that the host, router or other device is aware of other than the entire network. This is an example of a "connectless oriented" protocol. A data collector 52 can be disposed in the network in a path between the entities "A" and "B", such that the data collector 52 **monitors** some of the packets that comprise a flow between "A" and "B." As a single point **monitor,** the data collector has no concept that there are two ends communicating. The data collector 52 identifies these entities "A" and "B" in various NARs produced by the data collector 52. At later stage in the processing, either in the data collector 52 or elsewhere in the accounting process 14 the NARs are correlated so that the NARs or some aggregated NAR produced by the data collector 52 or the rest of the accounting process 14 can be associated with the accountable entities "A" and "B" to thus identify a connection between entities "A" and "B."

Detail Description Paragraph - DETX (29):

[0056] Thus, the data collector 52 is a single point **monitor, that monitors** traffic at one point in the network and converts the traffic into a "pipe oriented" or "flow oriented" accounting information. The data collector 52 identifies a source and a destination of the traffic. That is, the data collector develops a "connection oriented tracking." By distributing data collectors 52a-52g (FIG. 2) through out the network the network can be modeled as pipes having two endpoints. A data collector can be disposed in a partial pipe. The data collector 52 determines that one end of the pipe refers to "A" and the end of the pipe refers to "B." The data collector 52 can be disposed anywhere along the network.

Detail Description Paragraph - DETX (32):

[0059] Referring now to FIG. 6, an example of data flow 130 through the accounting process 14 is shown. In this example, data flow is initiated by a user 131 making a call to a remote access concentrator (RAC) 132. Upon receiving the call, the RAC 132 authenticates the user against a secure access controller 134. After verification, the RAC 132 connects the user to the network 135 and sends a RADIUS Start record (not shown) to the accounting process 14. The accounting process 14 generates a RADIUS Start NAR 137a and stores the RADIUS start NAR in a **database** 62. At that point, the remote user may check e-mail, look at a web server and transfer a file. For each transaction, the accounting process 14 captures the IP traffic, generating a e-mail, http, and ftp network accounting records 137b-137d, respectively. These are stored in the **database** 62. Upon completion of these transactions the user would log out of the network, at which time the RAC would send the accounting process 14 a RADIUS Stop record. The accounting process 14 generates a RADIUS Stop NAR 137e and stores the RADIUS stop NAR in the **database** 62. All of these records reflecting the user's transactions could be viewed and **reported** in flexible ways dependent on the needs of an end-user application.

Detail Description Paragraph - DETX (39):

[0066] The summary NAR and activity NAR have a one-to-many relationship. That is, while there can be a single summary NAR for a particular user over a particular call that will contain information about the sum of usage of network **resources** over the duration of the call, there can be many activity NARs. The activity NARs capture details about the actual activity and applications being used during the call. The summary NAR, therefore, depicts the total expense of the transaction or a set of transactions on a network, whereas, the activity NARs depict expenses of a transaction at any point in time. The summary NAR is generated in the flow aggregation process 60, as will be described below. In essence, the summary NAR is generated from individual activity NARs correlated in the data collectors 52a-52g, as will be described below.

Detail Description Paragraph - DETX (45):

[0072] The Network Accounting Record Identifier 202 acts as a **database** key value that makes the NAR 200 unique within the entire accounting process 14. The Network Accounting Record Identifier 202 allows the NARs to be handled and managed using **database** functions such as **database** integrity analysis and reliability analysis. The Network Accounting Record Identifier 202 also gives the accounting process 14 the ability to track the source of NARs and to build mechanisms such that the accounting process 14 can maintain identity of the origination of NARs throughout the system 10.

Detail Description Paragraph - DETX (48):

[0075] Thus, the NAR Identifier 202 provides **database** constructs to a NAR, whereas, the plurality of Network Accounting Record Attributes 204a-204n provide the actual metrics used for network activity **reporting** and network accounting.

Detail Description Paragraph - DETX (49):

[0076] As shown in FIG. 8B, the Network Accounting Record Identifier 202 (NAR_ID) is a set of objects within the NAR that uniquely identifies the NAR throughout the accounting process 14. The NAR_ID 202 is designed to support a number of properties of a NAR including flexibility, accountability, reliability and uniqueness. In order to provide these properties, the NAR_ID 202 is divided into objects designed to specifically provide these properties. Flexibility is supported through a NAR_HDR 203 section of the NAR_ID. Accountability is attained in the NAR through explicit identification of the source of the NAR by a component identification NAR_SRC_ID 203a. The source time is maintained in a NAR_SRC_TIME 203b. Reliability is supported, as described above, through the use of a NAR sequence number (NAR_SEQ_NUM) 203c, which is designed to enable traditional **database** integrity mechanisms.

Detail Description Paragraph - DETX (50):

[0077] The NAR_ID 202 is used to provide uniqueness for each NAR. The responsibility for guaranteeing the uniqueness of each NAR is handled by every accounting process component that has the ability to originate/source network accounting records. This responsibility requires that each accounting process component have the ability to unambiguously identify itself in each NAR that it produces. Thus, NAR type identifier, NAR_TYPE, is comprised of the source component identifier, NAR_SRC_ID, the NAR source time, NAR_SRC_TIME, and the NAR sequence number, NAR_SEQ_NUM. These three data objects act as a **database** key for a particular network activity record, ensuring the uniqueness of the NAR throughout the entire system.

Detail Description Paragraph - DETX (51):

[0078] The NAR_SEQ_NUM can have several purposes. One way that the NAR_SEQ_NUM can be used is as a discriminator when two NARs are produced at the same time. A second way that the NAR_SEQ_NUM is used is as a monotonically increasing index to ensure **database** integrity. Because the NAR_ID is unique, it should be considered as an allocated value. A NAR_ID is allocated at NAR origination.

Detail Description Paragraph - DETX (65):

[0092] Referring now to FIG. 11B, a NAR_USER_ID data structure 230 is the general type for identifying an accountable user. The accounting process 14 can use any available object type to represent the NAR_USER_ID value 232. The NAR_USER_ID value 232 will be a system established STRING type or a user index as generally supplied by a **database** system. The semantics of the NAR_USER_ID value 232 are consistent within the accounting process 14, and can be consistent outside of the accounting process 14.

Detail Description Paragraph - DETX (68):

[0095] Referring now to FIG. 11E, a NAR_FLOW_DESC data structure 260 is the general type for **reporting** on flow based network activity. The NAR_FLOW_DESC is a composite data structure 260 including a IP Source Address 262, IP

Destination Address 263, Transport Protocol 264, Type of Service 265, Source Port 266 and Destination Port 267 that are populated from transport and network layer of IP packets via flow probe. The NAR_FLOW_DESC NAR_ATTR Qualifier provides for Role designation, indicating whether the object referenced is acting as a source, destination, both or undeterminable within the system. These bits are set when the Role can be determined without ambiguity.

Detail Description Paragraph - DETX (70):

[0097] A set of accountable entities includes Username and Network Object Identifiers. There can be additional descriptive information available within network activity **reports** and within networking components that could be used to further describe accountable entities. These entity attribute descriptors can be used in the accounting process 14 to provide additional flexibility in how network activity information is **reported** and tallied. Support for entity descriptions can include object support for:

Detail Description Paragraph - DETX (84):

[0111] Prior to commencing transfer, the flow data collector 52 determines 350 if the flow aggregation processor 60 is available to receive NARs. If the flow aggregation processor 60 is unavailable, the flow data collector stores 352 the NARs to be transferred in its local store 314 (FIG. 16). The flow data collector 52 continues to check 354 the availability of the flow aggregation processor at periodic intervals until the connection between the flow aggregation processor 60 and the flow data collector is re-established. When the periodic status check indicates 350 that the flow aggregation processor is available, the flow data collector loads 356 NARs into the flow aggregation processor 60. The loading function can be implemented according to one of many strategies, e.g., a **database,** file, or data streaming strategy. Other strategies could be used. When the flow data collector receives 358 a confirmation or acknowledgment back from the flow aggregation processor that the NARs were loaded, the transfer is deemed successful and the locally stored copies of the transferred NARs are removed 360 from the local store. Thus, the "store and forward" capabilities of the flow data collector provide a **measure** of fault tolerance at this accounting process level to ensure reliable data transfer. The flow data collector only transfers NARs when it has determined that the flow aggregation processor is available and it considers the NAR transfer successful only upon receipt of an acknowledgment from the flow aggregation processor.

Detail Description Paragraph - DETX (87):

[0114] Referring now to FIG. 16, one implementation of the FAP 60 is as a **database** management system, or more specifically, a Structured Query Language (SQL) **database** management system, like those commercially available from Oracle or Sybase. Although not shown, it will be appreciated that the FAP is installed on a computer system, such as a host computer. Implemented as a **database** management system, the FAP includes a **database** server 400 coupled to a **database** 402. The FDCs 52 (from FIG. 14) can use the "push" model to move NARs up to the FAP via SQL calls. The **database** 402 stores a plurality of tables 404, including a NAR table 406 (implemented as a persistent cache) and an aggregation store 408. Also stored in the **database** are a plurality of SQL

commands and procedures (functions) 410 to be executed by the server 400. The functions include a FAP correlator 412, a FAP enhancer (enhancement process) 414 and a FAP aggregator 416. The **database** also stores a configuration file 420 for storing configuration parameters such as time and policy information. The operation of the FAP will be described below with reference to FIG. 17.

Detail Description Paragraph - DETX (88):

[0115] Referring to FIG. 17, an overall flow aggregation process 430 performed by the FAP is shown. The FAP receives 432 a NAR from one or more FDCs and loads 434 the received NAR into a persistent store or cache (of **database** 492 from FIG. 16). If the FAP is unable to load the NAR, it requests 436 that the transferring FDC resend the NAR. If the load is successful, the FAP sends 438 an acknowledgment back to the sending FDC. The FAP determines 440 if the NAR can be correlated (with or without enhancement). If the FAP determines that the NAR can be correlated, the FAP correlates 442 the NAR with other NARs received from other FDCs. Once the NAR is correlated, it may be enhanced 444 "across the system", in a manner more fully described with reference to FIG. 18. The NAR may be enhanced 446 to include enhancement information obtained from an outside source (i.e., collected by a data collector for a different equipment interface). Once any potential correlation and enhancement has been performed, the FAP determines 448 if the NAR is a candidate for aggregation. If so, the FAP applies 450 the aggregation policy 420 (from FIG. 16) and stores 452 the resulting aggregated NAR in the aggregation store until a predetermined time expires or event occurs 454 (as set in the FAP configuration 420). The FAP ensures 456 the uniqueness and integrity of any NAR by examining NAR header information prior to re-loading 458 such NAR into the persistent store.

Detail Description Paragraph - DETX (89):

[0116] The accounting architecture may be implemented to include a second "shadow" FAP process, also coupled to the data collectors and operating in the manner described above with respect to receiving and processing NARs. In the dual/shadowing FAP implementation, the accounting architecture further includes an error detection module (not shown) coupled to both of the first (primary) and second (shadow) FAP processes. The error detection module operates to detect an error relating to the first flow aggregation process and cause the aggregate **reports** from the second flow aggregation process to be transferred to the accounting module (i.e., flow data distributor 70) in place of the aggregate **reports** from the first flow aggregation process.

Detail Description Paragraph - DETX (94):

[0121] Still referring to FIG. 18, the NAR1 508 has an IP-to-username mapping 512 and an accounting interval 516 comprising a start time and a session time to indicate a time interval bounded by start time "T1" and a start time+session time ("T2"), that is, the accounting interval represents a start time and a stop time. The username 524 in the IP address-to-username mapping is supplied by the DHCP server 500. In the FAP, this NAR1 information will either go directly to a correlation function or to the local store (which could either be a **database,** file or memory), where it can be directly accessed by the correlator function. The NAR2 510 has an accounting entity ID 514, a T3-to-T4

accounting time interval 518 and a metric 530. The accounting entity identifier 514 has two IP addresses 526, 528, one corresponding to a source IP address and the other corresponding to a destination IP address. The NAR2 502 is passed to the correlator 442, which determines that the T1-to-T2 time interval 516 from the IP-to-username address map in the NAR1 508 overlaps or in some way relates to the T3-to-T4 time interval 518 of the NAR2 510. The correlator determines that T1, T2, T3 and T4 are related, and that the IP address 522 in the IP-to-username address mapping 512 is associated with one of the two IP addresses 526, 528 in the NAR2 510. Thus, the FAP enhances the NAR2 510 by inserting information from the accounting entity ID 512 (of NAR1 508) into the accounting entity ID portion of the NAR2 510. The resulting, enhanced NAR2 532 has an enhanced accounting entity ID 534 that includes the T3-to-T4 timestamp (not shown), the IP-to-IP addresses 526-528 and the username 524. Thus, the enhanced NAR2 now has a mapping between the username and the one of the IP addresses 526, 528 that is related to the IP address 522. The metric 530 is unchanged.

Detail Description Paragraph - DETX (97):

[0124] Referring again to FIG. 19, another correlation and enhancement process 442, 444 maps the username 524 to a workgroup. The FAP builds up search keys using **database** principles and relational algebra. Thus, for example, the IP address has a one-to-one mapping with a username. (The one-to-one mapping is assured because of the nature of IP addressing and the way that the DHCP server assigns usernames.) Therefore, there can be only one user for an IP address in a given instance. These terms or values are equivalent keys, so the username can easily be replaced with the IP address. The username 524 that was inserted into the enhanced NAR2 532 can be used as a look-up into a workgroup 540 in one of the **database** tables 404 (FIG. 16) because the user is actually a member of a workgroup. Therefore, the enhancement function can be used to insert the workgroup label into the enhanced NAR2 (already enhanced for username) to produce a twice-enhanced NAR2 542. If the now twice-enhanced record 542 is to be aggregated, it is held in the aggregation store 408 (FIG. 16) for some time period T until other NARs are received for potential aggregation.

Detail Description Paragraph - DETX (114):

[0141] As discussed above in reference to FIG. 2, the accounting process supports a flow probe e.g., 12c that captures a user's network activity for purposes of IP accounting. The flow probe 12c **monitors** all traffic over a given network link and captures data associated with the different flows in the traffic on that link. It is capable of **monitoring** IP data flows over a number of technologies (e.g., Ethernet, ATM, FDDI, etc.).

Detail Description Paragraph - DETX (115):

[0142] One important feature of the flow probe is its ability to detect and **report** on successful and unsuccessful connectivity. This capability is useful to billing and chargeback applications. For example, a user may try to connect to a particular switch or reach a particular network, but is rejected. The flow probe 12c can identify that transaction as unsuccessful and provides the billing application with information that the billing application can use in

determining whether or not the user should be charged for that transaction. The flow-based connectivity model embodied in the flow probe is described generally with reference to FIGS. 23-25, and specifically with reference to FIGS. 27-28.

Detail Description Paragraph - DETX (120):

[0147] In a given network segment **monitored** by the flow probe, much of the typical IP traffic includes TCP protocol traffic. Because the flow probe is a flow based **monitor** that is actually tracking the TCP as a flow, it is completely aware of the TCP protocol and that protocol's three-way handshake algorithm (state machine). The TCP flow has indicators to indicate that a connection is being established or a flow is being disconnected. However, these messages are only relevant to the two communicating parties (e.g., A and B in FIG. 27). The end system A may request that it be able to communicate with B and sends a "TCP SYN" indication. Any of the networking devices 608 along the path 606 can reject this SYN request, completely independent of the intended destination (in this example, end system B) and without the knowledge that the end system B is a party to this communication request. There are a variety of problems that can cause an internal network component to reject a request. For example, a router between A and B may find that there is no route available for forwarding a packet towards B or that the routing path is inoperable (and no alternate exits), or the router may find that it doesn't have the **resources** to handle the packet.

Detail Description Paragraph - DETX (122):

[0149] As an independent **monitor** operating outside of the context of the originating entity ("A", in this example), the flow probe is able to produce a complete and accurate record of the transaction by mapping the network control information to the user request information. To do so, flow probe correlates the state information in protocols such as TCP with error event or condition messages provided by other protocols, such as ICMP. In this manner, it is possible to determine if a particular request for a service has actually been denied as a result of some network independent event. The flow probe correlates the dissimilar protocols together and finds a way of representing the network event in its normal **reporting** of the TCP flow.

Detail Description Paragraph - DETX (123):

[0150] The flow probe has specific **reporting** mechanisms for the specific protocols. The TCP protocol, for instance, has many more metrics associated with its protocol states than UDP based flows. However, because ICMP relevant events or network relevant events are not associated with or have any impact on the state of TCP or UDP or any of the normal protocols, the flow probe provides a mechanism for tagging its state tracking with the error event. The NAR is represented as a start flow indication, a continuing or status record and a stop record. All of the flow 15 probe's internal protocol indications map to start, continuous or stop states. When a network rejection event comes in (e.g., in the form of an ICMP message, or other type of internet control information), regardless of what state the probe is tracking as the current state, it reverts to a stop state and has to expand upon the normal time or transition based stop conditions to include an specific ICMP event as the cause

of the closed state. The flow probe NAR includes bit indications for the actual protocol states that it is tracking. For ICMP generated events, the flow probe indicates whether the source or the destination was affected by the events. In order to convey this network rejection or network event back to the parent flow, the NAR allows for specific network rejection logic to be **reported** either by the source or the destination, and has specific bit indicators in either the source or the destination fields.

Detail Description Paragraph - DETX (124):

[0151] There are two key aspects to the connectivity scheme of the flow probe as described thus far. First, the probe determines that an ICMP event has occurred. Second, the probe correlates that event to the "parent" flow, i.e., the same flow as that associated with the failed request, and stores the exact ICMP event into some state associated with that flow so the event can be **reported** to the accounting system in a NAR. At this point it may be useful to examine the IP packet and ICMP message formats in general, as well as examine certain fields of interest.

Detail Description Paragraph - DETX (126):

[0153] Referring to FIG. 26, an exemplary ICMP message format 622 for **reporting** errors is shown. The format includes an ICMP message header 624. The header 624 includes a type field 630, which defines the meaning of the message as well as its format, and a code field 632 that further defines the message (error event). The error **reporting** message types (type values) include: destination unreachable (3); source quench (4); source route failed (5); network unreachable for type of service (11); and parameters problem (12). Each of the types has a number of code values. For a destination unreachable message (TYPE field value is 3), the possible codes (code values) include: network unreachable (0); host unreachable (1); protocol unreachable (2); port unreachable (3); fragmentation needed and DF set (4); source route failed (5); destination network unknown (6); destination host unknown (7); source host isolated (8); communication with destination network administratively prohibited (9); communication with destination host administratively prohibited (10); network unreachable for type of service (11); and host unreachable for type of service (12).

Detail Description Paragraph - DETX (128):

[0155] It will be understood that TCP is an example protocol. The field 636 could correspond to a portion of packet header from a packet of another protocol type. Also, the error **reporting** protocol could be a protocol other than ICMP, and the amount of header in field 636 could be more or less than 64 bits, that is, this amount may be adjusted so that the appropriate flow information can be obtained from the header of the message contained in the discarded IP packet, as described below.

Detail Description Paragraph - DETX (129):

[0156] Referring to FIG. 27, a packet processing method ("the process") 650 performed by the flow probe is shown. The process captures 652 a new IP packet (datagram) and tests 654 the received packet to determine if it is good (i.e.,

well-formed). The process 650 examines 656 the protocol field in the IP packet header to determine if the protocol is the ICMP protocol. If the protocol is ICMP and the information type field is set to one of the five error **reporting** messages described above, the process bypasses the IP packet and ICMP message headers and processes 658 the ICMP message or packet payload (FIG. 26), which corresponds to a portion of IP packet which that was discarded and to which the event message relates. The payload process will be described with reference to FIG. 28 below. Once the payload processing is complete, the processing of the IP packet resumes 659 the processing that would be performed if the IP packet had not been detected as containing an ICMP message of the error **reporting** variety as discussed above, as will now be described.

Detail Description Paragraph - DETX (130):

[0157] Still referring to FIG. 27, if the protocol is not ICMP and/or the information type is not an error **report,** the IP packet is processed as follows. The probe scans 660 the header to determine the values of the fields which correspond to the "flow key", the fields which define "the flown for the probe. Each flow probe can be configured for a particular flow key definition. For example, the flow key might be the source/destination IP addresses, the source/destination ports and the protocol. The probe determines 662 if the flow key of the processed packet header matches a flow already stored in the flow probe. A local store in the flow probe is used to hold flow representations including flow key parameters, metrics, state information. The state information will include, in addition to the protocol control-related states (i.e., TCP "FIN"), error event/state change cause and source/destination to which the message is addressed. These flow representations are converted into NARs for accounting process **reporting** purposes.

Detail Description Paragraph - DETX (133):

[0160] Thus, the payload processing can be viewed as a packet processing exception, an exception that is invoked when it is determined that an ICMP error **reporting** message has been received. The ICMP messsage **reports** a error event and the IP packet associated with that error event. The exception process serves to correlate the flow of the discarded IP packet in the ICMP message with the parent (matching stored) flow, thus mapping the ICMP error (state) information to the parent IP flow.

Detail Description Paragraph - DETX (134):

[0161] The flow probe **reports** on network traffic activity through a flow probe NAR, which **reports** IP flow traffic activity. The flow probe categorizes network traffic into one of four classes of traffic flow: I) connection oriented (e.g., TCP); ii) new connectionless; iii) request/response connectionless (e.g., UDP, DNS); and iii) connectionless persistent (e.g., NFS, Multicast BackBONE or "MBONE" multicast traffic). To each of these class it applies connection oriented semantics for a uniform approach to status **reporting**. That is, flow probe treats these dissimilar transaction models as if they were the same. There is one uniform structure for the status **reports** generated for each of the 4 different transactions. Each status **report** includes transaction start and stop information, MAC and IP source and destination addresses, the IP options that were seen, the upper layer protocol

used, the transaction source and destination byte and packet counts and upper layer protocol specific information. The protocol specific information and the criteria for when the status **reports** are created, is different for each of the four transaction types.

Detail Description Paragraph - DETX (135):

[0162] The connection oriented protocol understood by the flow probe is TCP. Flow probe has complete knowledge of the TCP state machine and thus can generate status **reports** with each state transition seen within any individual TCP. There is also a provision for generating time interval based status **reporting** in the TCP connections that the flow probe is tracking. The status **report** indicates which states were seen, if any packets were retransmitted, if the source or destination had closed, and if the **report** had been generated by a time condition. In a default mode, the flow probe generates a cumulative status at the time a TCP closes, or times out. This strategy offers the greatest amount of data reduction on transactions.

Detail Description Paragraph - DETX (136):

[0163] Any non-TCP traffic is categorized as a connection-less transaction. When configured to generate the most detailed level of **reporting** for connectionless traffic, the flow probe can **report** the discovery of a new connection-less transaction; the existence of a request/response pair within the transaction (as exists when the probe has seen a single packet from both the source and the destination for the transaction); the continuation or transaction persistence, and so forth. The transaction persistence status is generated with a timer function. If it has been seen within a configured timer window, a **report** is generated.

Detail Description Paragraph - DETX (137):

[0164] The status **report** for non-TCP traffic indicates if the **report** is an initial **report,** a request/status **report** or a continuation (or a current transaction) **report**.

Detail Description Paragraph - DETX (138):

[0165] In the default mode, the flow probe generates a status **report** when it has seen a request/response "volley" within a transaction and every 15 minutes thereafter, if the transaction persists. This offer immediate notification of request/response traffic and a fair amount of data reduction on connection-less transactions.

Detail Description Paragraph - DETX (139):

[0166] Thus, the flow probe state tracking includes protocol- specific state information. It provides detailed information on transport specific flow initiation, such as TCP connection establishment, as well as flow continuation and termination event **reporting**.

Detail Description Paragraph - DETX (140):

[0167] Protocol Independent Packet **Monitor**

Detail Description Paragraph - DETX (141):

[0168] Referring to FIG. 29A, a network 700 includes a **monitor** 702 that runs a process for detecting packet loss. The **monitor** 702 will be particularly described using IP SEC authentication headers. The **monitor** 702 uses sequence numbers that exist in IP SEC authentication headers. The **monitor** 702 can be used to detect lost packets in any type of protocol that uses sequence numbers in headers of the packets, etc. The **monitor** 702 is an independent **monitor** that can be disposed anywhere in the network 700. The **monitor** 702 is protocol independent.

Detail Description Paragraph - DETX (142):

[0169] The network 700 would include a plurality of such independent **monitors** 702 each disposed at corresponding single points in the network 70. Typically, the **monitor** 702 can be disposed in-line such as in a network device such as a switch, router, access concentrator, and so forth. Alternatively, the **monitor** can be disposed in an out of line arrangement in which network packets are copied from the device and coupled to the out-of line **monitor**.

Detail Description Paragraph - DETX (143):

[0170] The **monitor** 702 examines each packet of a network flow that passes through the device associated with the **monitor** 702. The **monitor** 702 receives serialized IP packets. The packets can have the format specified by the Network Working Group, by S.

Detail Description Paragraph - DETX (145):

[0172] Referring now to FIG. 29B, a packet loss detector process 704 that runs in the **monitor** 702 is shown. The packet loss detector process 704 examines 706 header information in the packet, to determine if the packet includes an authentication header. If the packet does not include an authentication header, then the packet loss detector process 704 ignores 24 the packet and exits to wait for the next packet. If the packet includes an authentication header, the packet loss detector process 20 tests 708 to determine if the packet loss detector process 20 had been tracking the flow that is represented by the source and destination IP addresses and the SPID value that is in the authentication header. The packet loss detector will perform a cache look up to determine if the flow is stored in a cache of currently tracked flows. The packet loss detector process 20 tests 708 those values to see if the packet loss detector process 704 is currently tracking that security flow.

Detail Description Paragraph - DETX (146):

[0173] If the packet loss detector process 704 is not tracking that security flow, the packet loss detector process 20 will establish 710 a flow cache entry for that flow in a cache that can be maintained in memory (not shown). The packet loss detector process 704 will store the source and destination IP address and the SPID value from of the authentication header. The flow cache

also includes all other authentication headers from other security flows that have previously been tracked. The flow cache enables the packet loss detector process 20 to **monitor** and track many hundreds, thousands, and so forth of different security flows. A cache entry is established for every different flow. Once the cache entry is established, the packet loss detector process 704 updates 712 the sequence number entry in the cache for that security flow. That is, the initial sequence number in the authentication header for the encountered flow is stored. The sequence number can start at any arbitrary value.

Detail Description Paragraph - DETX (150):

[0177] When packets may traverse more than one packet **monitor** 10, the packet loss detector process 704 may produce a packet loss detected indication that does not indicate that the packets were actually dropped. A packet loss drop indication in a multi-**monitor** embodiment indicates that the lost packets did not come through the particular packet loss detector process 704. However, the indicated lost packets could be on other segments of the network. That is, it is possible that other parts of the current flow are in other parts of the network. Therefore, the packet loss detector process 704 notes how many packets were actually successfully transmitted, as well as lost, and optionally their sequence numbers. These values can be compared to other values from other **monitors** 702 to establish whether or not there had been packet loss for the flow through the network.

Detail Description Paragraph - DETX (151):

[0178] This indication, could be converted into Network Accounting Records thus would be coupled to a process e.g. the accounting process 14 that **reports** statistics on that particular flow to provide a summary of how many packets were lost relative to how many packets were actually successfully transmitted on the flow. In the accounting process 14, the network accounting records are correlated, aggregated, enhanced and so forth to identify network flows. This information can be used to determine the records that correspond to a particular network flow and whether a determined network flow lost any packets.

Detail Description Paragraph - DETX (155):

[0182] By deploying the accounting process 14 to observe service quality, the capturing quality of service process 730 can validate performance of **service level agreements** (not shown). If the capturing quality of service process 730 detects that the policy level specified in a **service level agreement** is not being enforced, then the policy can be reassessed, redefined, and redeployed 742. The capturing quality of service process 730 can again observe 737. Through the observation 736, the capturing quality of service process 730 can determine whether reassessment and redefining of the deployed policy was successful. Several cycles of this quality of service optimization process could be required.

Detail Description Paragraph - DETX (156):

[0183] An important component of quality of service includes determining whether there has been packet loss. The packet detector **monitor** described in

conjunction with FIGS. 29A and 29B can be used to access packet loss. The packet detection **monitor** 702 can be deployed in the network and generate NARs that can be used to determine packet loss as discussed above. This information can be used in the capturing quality of service process 730 to assess whether the policy specified by the **service level agreement** was provided to the customer. Additionally, so called Differentiated Service "DivServe technology" that a known quality of service solution that has been proposed for the Internet as well as enterprise networks. In contrast to a per-flow orientation of some types of quality of service solutions such as Int-serv and RSVP, DiffServ enabled networks classify packets into one of a small number of aggregated flows or "classes", based on bits set in the type of service (TOS) field of each packet's IP header. This is a quality of service technology for IP networking is designed to lower the statistical probability of packet loss of specific flows. The capturing quality of service process 730 establishes DivServ policy, that is decomposed into a collection of DivServ configurations. The DivServ configurations are deployed to a collection of routers or switches that the customer would have access to in the network 11 as part of the enforcement/deployment process 732. Because packet loss is a statistical phenomenon, the capturing quality of service process 730 observes 736 a large number of network flows. The capturing quality of service process 730 can observe network traffic because of the use of the accounting process 14 and the resulting NARs at the granularity in which the DivServe policies are actually being deployed. The DivServe policies are generally deployed at the source and destination IP address, protocol and possibly destination port level.

Detail Description Paragraph - DETX (158):

[0185] As mentioned, because IP network quality of service is a statistical phenomenon, the capturing quality of service process 730 obtains a large number of samples, over a long period of time. Through this **optimizing** capturing quality of service process 730 and DivServe deployment 734, the customer will get beneficial policy deployment for this service.

Detail Description Paragraph - DETX (161):

[0188] A service management feedback process 750 therefore includes three components, service provisioning 752, policy server 754 and service accounting 756. The role of service provisioning 752 is to send requests 752b to the policy server 754 to obtain an appropriate active policy, and obtaining rules and domain information 754a from the policy server. The provisioning system can communicate with appropriate network management systems and element management systems (not shown) to configure the network 10 for an end-to-end service. When the configuration 752a is deployed at the various network devices (not shown) at that point, the service is produced. The level of service is **monitored** or audited by the accounting system 756 which can be the accounting process 14 described above. The accounting process 14 **monitors** the level of service by producing appropiate newtork accounting records. The newtwork accounting records NARs are used by a billing application to adjust billing based on the level of service that was provided as determined by the accounting system 14. The accounting system 14 also can compare the policies produced by the policy server to the actual levels of service provided to the customer by examining NARs that are produced by the customer's usage of the network.

Detail Description Paragraph - DETX (165):

[0192] In that case, the provisioning service configures 752 the policy enforcement mechanism that was put into the router in the network. How the policy was defined to the provisioning equipment is that there is a one-to-one relationship between the policy and what the accounting process 14 will **monitor** in the network. The accounting process 14 will be aware that company "X" contracted to have 100% availability from the router.

Detail Description Paragraph - DETX (167):

[0194] Capturing quality of service as audited by the accounting process 14 includes detecting of packet loss, as mentioned above. Each of the components managed by the service management process 750 require information. Therefore, the service provisioning has to provision these various quality levels. The policy server 754 thus, keeps what is essentially enforcement of the levels of quality that are offered by different service types, and the accounting process 756 detects, **monitors** and audits whether those classes in quality of service are being delivered.

Claims Text - CLTX (3):

2. The method of claim 1 wherein observing further comprises: determining at the network that **resources** are not available for providing the first level of service; and, in response to said determination, providing a second level service.

Claims Text - CLTX (6):

5. The method of claim 1 further comprising: determining whether there has been packet loss; and wherein determining packet loss includes: deploying a packet detector **monitor** in the network to generate network accounting records that can be used to determine packet loss.